Journal of Theoretical and Applied Information Technology
15th June 2018. Vol.96. No 11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A STUDY ON DYNAMIC HAND GESTURE RECOGNITION FOR FINGER DISABILITY USING MULTI-LAYER NEURAL NETWORK

[1]ROMI FADILLAH RAHMAT, [2]SARAH PURNAMAWATI, [3]EKA PRATIWI GOENFI, [4]OPIM SALIM SITOMPUL, [5]MUHAMMAD FERMI PASHA, [6]RAHMAT BUDIARTO

[1,2,3,4]Departement of Information Technology, Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Medan, Indonesia

[5]School of Information Technology, Monash University Malaysia Campus, Malaysia

[6]Department of Information System, College of Computer Science and Information Technology, Albaha University, Saudi Arabia

E-mail: [1]romi.fadillah@usu.ac.id, [2]sarah_purnamawati@usu.ac.id

ABSTRACT

Interaction between human and computer is generally performed with a keyboard and mouse. However, these interactions have certain drawbacks which cannot be done by users with physical disabilities or user who have disability from the wrist to the fingertip. To overcome this problem, an approach to recognize human hand gesture as a means of human-computer interaction is needed. The method proposed by the author is the use of algorithms: nearest neighbor, grayscaling, frame-differencing, Principal Component Analysis (PCA) and Multi-Layer Perceptron (MLP). This research was conducted in two experiments, which were experiment with six different types of hand gestures and experiments with four different types of hand gestures. Each experiment was performed five times with different value of number of hidden layers parameter and hidden neurons parameter. The best testing result obtained from the experiment with six types of hand gestures is from the second experiment with two hidden layers using 300 and 50 hidden neurons for each layer, resulting in an accuracy rate of 77.02%. The best testing result obtained from the experiment with four different types of hand gestures is from the first experiment with two hidden layers using 300 and 50 hidden neurons for each layer, resulting in an accuracy rate of 89.72%. The best overall result is then implemented into the front-end system for controlling application such as: file explorer, music player, video player, slideshows and PDF reader.

Keywords : *Dynamic Hand Gesture, Multilayer Perceptron, Finger Disability, Image Processing*

1. INTRODUCTION

One of the most common ways to interact with a computer is by using keyboard and mouse. However, this interaction can't be done by a handicapped person; handicapped from the wrist to the fingertip or amputated wrists or fingertips. To cope with this problem, an approach for hand gesture recognition is needed by ignoring the fact that a person is either normal or handicapped as a tool for human and computer interaction.

Research about hand gesture recognition have been done before, research about static hand placement in real-time and continued with implementing Artificial Neural Network (ANN) method [1], another research tried to implement hand gesture detection and recognition dynamically in real-time for human and computer interaction [2] and another about hand gesture recognition statically using Hue Saturation – Chroma Blue Chroma Red (Hs-CbCr) [3].

The previous researches have some weakness, that is, the accuracy will decrease when the hand gesture recognition is done in large quantities. The recognition process is still static so that it is less interactive. In the research about dynamic hand gesture recognition, it was found that there was a big chance that images caught

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

were not the same as those already saved in the database. To solve these flaws, we tries to research the problem by implementing Multi Layer Perceptron (MLP) method in the human and computer hand gesture recognition. The research question is how effective MLP in order to solve dynamic hand gesture recognition in VIVA dataset to be used in finger disability application.

MLP is feed-*forward Neural Network* with more than one hidden layer. MLP is a machine learning technique which use a large number of data to train the computer to do what usually humans do like recognizing images, texts and languages while sorting it as well. MLP uses many layers from non-linear information processing to do extraction and forming controlled features, as well as for pattern analysis and classification [4].

Human and computer interaction is normally done by using keyboard and mouse. This interaction cannot be done by a handicapped person. Handicapped from the wrist to the fingertip or amputated wrists or fingertips. To cope with this problem, an approach for hand gesture recognition is needed by ignoring the fact that a person is either normal or handicapped as a tool for human and computer interaction.

Researches about human hand gesture recognition and *Deep Neural Network* method have already been done before by other researchers. Neto, et al. conducted a research about placement for static hand gesture in real-time and continued it by using *Artificial Neural Network (ANN).* The result of the research showed that the recognition level was very good (99.8% for ten hand gestures and 96.3% for thirty hand gestures) [1]. The scope of this work is the limitation to only static hand gesture data that has been used not dynamic data.

Tang. et al. did a research about static hand posture recognition in real-time by using a special device called *Kinect.* The device will help the pre-processing one step ahead than traditional approach. In term of methodology, this research implemented *Deep Belief Network (DBN)* and *Convolutional Neural Network (CNN)* to recognize hand posture. The result of the research showed a good performance in the hand detection and tracking process (DBN: 98,063% and CNN: 93,995%) [5]. One of the limitation is the application will be very restricted to the device limitation. However DBN and CNN will give a good performance to classify the hand posture.

Erhan, et al. proposed a method to track an object's location in an image and mark it with a box marker one image at a time. The method used was *Deep Multi Box* which implemented *Deep Convolutional Neural Network* as a base for features extraction and model training to predict the location of the box marker. The accuracy generated when the method was implemented for detection was 58.48% and for classification was 77.29% [6].

Ramjan, et al. did a research about dynamic hand gesture detection and recognition in real-time by using human and computer interaction. The process carried out was image capturing from video, extracting objects on the hand in the background of the captured image as well as web camera usage for inspecting object's movement. The object caught was then processed with *blurring, grayscalling, Hue Saturation Value (HSV),* and *blob detection.* After these processes were done, the image would be matched with the image saved in the database to dertermine the movement [2]. The image processing process conducted in this research will give good detection process, however no machine learning used in this research that will limiting the variation of input from the user.

Rahmat did research about static hand gesture recognition by implementing *computer vision* technique, *Hue Saturation – Chroma Blue Chroma Red (*Hs-CbCr) to detect skin colour and *average background* technique to handle the *background* problem. The accuracy level in recognizing the human hand gesture was 96.87% on condition that sufficient lighting of the image was fulfilled [3]. The research shows promising result in skin detection for static hand gesture. The method used will give benefit in order to create the static hand gesture application.

Molchanov, et al. researched about dynamic hand gesture recognition while driving a car by using the *Convolutional Neural Network (CNN).* This research utilized the dataset from *Vision for Intelligent Vehicles and Applications (VIVA).* The result of the research generated an accuracy of 77.5% for classification process [7].

If we take a look at the previous research, it shows that Deep Neural Network, Convoloutional Neural Network and Deep Belief Network shows promising result regarding hand gesture recognition, however, Pedro [1] shows good performance of ANN in static hand recognition and static hand recognition using non classifer as shown in Rahmat [3], while this research in dynamic hand recognition. One of the

Journal of Theoretical and Applied Information Technology
15th June 2018. Vol.96. No 11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

closest and related research is done by Molchanov [7] using similar dataset and dynamic hand gestur recognition, with different type of movement compared to our research, he uses CNN and generate 77,5% in accuracy.

2. METHODOLOGY

The method proposed by the writer for human hand gesture recognition consists of some processes. The processes carried out can be described as follows : the taking of human hand gesture by using camera; grouping of hand gesture videos with the type of hand gestures that was saved; every video of hand gestures will then be separated into several image frame; every image will then be processed using *nearest neighbor* algorithm; *grayscalling* frame *frame differencing*; four images will be selected from a set of images which had gone through the *frame differencing* process, and these images will be processed again with with PCA; MLP is trained using a *training dataset* and is tested using *testing dataset;* MLP will then be implemented to manage some applications such as *file explorer, music player, video player, slideshow* and *PDF reader.* The general architecture which describes the methodology used in this research can be seen in Figure 1.

2.1. Dataset

Data used in this research are collected from *VIVA dataset* (*Vision for Intelligent Vehicles and Applications*), which is sourced from http://cvrr.ucsd.edu/vivachallenge/index.php/hands/hand-gestures/. There are two experiments which are carried out in this research:

1. First experiment
   The *dataset* used in the first experiment consists of six category of human hand gesture videos such as left to right gesture, right to left gesture, top to bottom gesture, bottom to top gesture, front to back gesture, back to front gesture with a total of 818 videos collected from this experiment.
2. Second experiment
   The *dataset* used in the second experiment consists of only four category of human hand gesture videos which is left to right gesture, right to left gesture, top to bottom gesture, bottom to top gesture with a total of 542 videos collected from this experiment.

The dataset used here is classified as waving movement hand gesture, which will not involving finger recognition. Collected videos are then divided into two datasets namely training dataset and testing dataset. Training dataset is utilized to train the MLP method and testing dataset is utilized to test the trained MLP method. All of the dataset used for training and testing is done randomly.

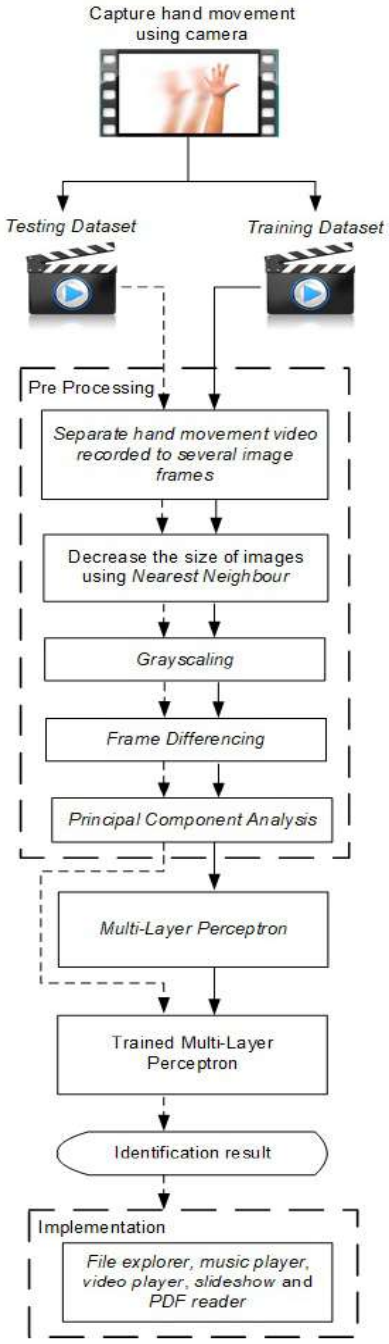


Figure 1. General Architecture

Journal of Theoretical and Applied Information Technology
15th June 2018. Vol.96. No 11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2.2. Separating the hand gesture videos into several image frames

Basically, a video is composed of many images. Therefore, every video which is recorded must be separated first into several frames before it can be used as an input for the MLP.

2.3. Shrinking the size of the image using nearest neighbor algorithm

Every image should be minimized into 50 x 23 pixel using the nearest neighbor algorithm. Nearest neighbor used the closest pixel to the first image to give its pixel value to image which is going to be enlarged or shrunk [8]. As for the purpose of shrinking the image size is to cut the total of parameters which will be used as the input for MLP so that it will accelerate the training process.

2.4. Grayscaling

After the image size is shrunk, it will be grayscalled. Grayscalling is the process of transforming colored image into gray image. Pixels which originally consists of RGB (Red Green Blue) will be transformed into one value which is gray through the grayscalling process. The equation for this process is:

$$Y = 0.299.R + 0.587.G + 0.114.B \quad (1)$$

Where: Y = converted image from RGB to Grayscale
R = the *red channel* value in the pixels
G = the *green channel* value in the pixels
B = the *blue channel* value in the pixels

The grayscalling process also aimed to reduce the number of parameters used by MLP so that the training process is accelerated.

2.5. Frame differencing

Every image that has gone through the previous stage will be processed with a method called *frame differencing*. Images will be separated with the first frame of it as a differentiator, intended to simplify images with irrelevant background. Four images will eventually be selected from the group of images that have undergone this process. This image selection process is done by choosing a lapse of image that has been differentiated and then divided by 5, where the first frame won't be used because it will be completely in black. This happens because the first frame and the frame differentiated have a common color which is black. The difference described is represented in equation 2, 3 and 4 [10].

Differential:

$$Dk = \begin{cases} f_k - f_{k-1} & f_k - f_{k-1} \geq 0 \\ 0 & \end{cases} \quad (2)$$

Negative Differential:

$$Dk = \begin{cases} |f_k - f_{k-1}| & f_k - f_{k-1} \geq 0 \\ 0 & \end{cases} \quad (3)$$

Fully Differential:

$$Dk = |f_k - f_{k-1}| \quad (4)$$

Where: Dk = image from *frame differencing*
f_k = *frame image*
f_{k-1} = *frame image* before the time interval

2.6. Principal Component Analysis (PCA)

After the previous process generated four images, it will go through the PCA process. This is implemented to minimize the dimentionality (pixel size) of the selected image so that it can speed up the training process. The first minimalization process will be carried out with six hand gestures and will reduce the pixel size from 4600 to 523, whereas for the second process with have four hand gestures and will reduce the pixel size from 4600 to 365. Every operation parameters PCA implemented in the training dataset will be saved to avoid redundancy in the training process. it is known that $X = \{x_n \in R^d \mid n = 1, 2, ..., N\}$ which represent a dataset with dimension d , from the dataset X , PCA will build dataset Z where $Z = \{z_n \in R^k \mid n = 1, 2, ..., N\}$ with dimension k, where k value is smaller than d. the steps held in the PCA includes [11] :

1. Normalize every dimension of data using the Z-score nomalization formula:

$$v'_i = \frac{v_i - \overline{A}}{\sigma_A} \quad (5)$$

After completing this process, the dataset will have a *mean* value of zero.

2. Find the covariance matrix using the following formula:

$$\Sigma = \frac{1}{m} X^T X \quad (6)$$

, where Σ is the covariance matrix; m is the amount of data in the real dataset; X is the real dataset which is arraged in the form of matrix with every column act as a feature,

and every row is a set of features for one object; X^T is the transpose of matrix X. the result of this covariance is matrix with the size of dxd, where d is the dimension from X.

3. Count the number of eigenvalue and the eigenvector from the covariance matrix, by doing the Single Value Decomposition to the covariance matrix. Eigenvalue and eigenvector are vectors which characterize dataset X. if the initial data with dimension d is available, then the eigenvector will be usable as much as d.
4. Sort the eigenvector likewise the eigenvalue in descending order, where eigenvector with eigenvalue with higher value on the left side of the matrix and eigenvectir with lower eigenvalue on the right side of the matrix.
5. Cast eigenvector as much as k to form feature vector U, where from the feature vector U will be formed data of PCA with dimension k. The selection of eigenvector is done based upon the value of eigenvalue from eigenvector. Eigenvector with the highest eigenvalue will be choosen, so that the primary data information will be kept safe from the PCA's result.

The definition of the value of dimension k for the dataset formed from PCA is driven based upon the retain rate. Retain rate is the percentage of information which is kept safe in the PCA's result from the real dataset. The designation of dimension k generally choosen by keeping the retain rate at 99%, which meet the requirement of equation 2.7. This is done so that all the primary dataset can keep be represented by the dataset from PCA.

$$\frac{\sum_{i=1}^{k} S_i}{\sum_{i=1}^{m} Si} \geq 0{,}99 \quad (7)$$

S_i is *eigenvalue* in position of i. all of the eigenvector's value which is choosen ten will be arranged as a column in a matrix with feature vector U.

6. The result of PCA dataset is produced using the following equation:

$$Z = U^T X \quad (8)$$

, where Z is the matrix of dataset from PCA; U is the matrix vector from choosen eigenvector which is obtained from the previous process; X is the matrix from the real dataset.

2.7. Implementation of Multi Layer Perceptron

Training dataset and testing dataset which have gone through all of the pre-processing will be used to train and test MLP. Implemented MLP which is actually Feed-forward Neural Network and trained using backpropagation method. Backpropagation method is one of algorithm which is used in the training of multilayer perceptron. As for the stages which is done in the algorithm can is [12] :

1. Initialitation
   Every weight that connects every neuron will be given random value and distributed equally and in small range (Haykin, 1999). Initialitation for every weight can be calculated with equation 9.

$$\left(-\frac{2{,}4}{F_i}, +\frac{2{,}4}{F_i} \right) \quad (9)$$

   Where : F_i = the number of input from neuron i in the network

2. Activation (*Feed-forward*)
   Activation process or feed-forward will insert all the existing input into the neural network to produce the output. The activation of the neural network is done by using input $x_1(p)$, $x_2(p)$, ..., $x_n(p)$ and the desired output is $y_{d1}(p)$, $y_{d2}(p)$, …, $y_{dn}(p)$, with p is the sum of iteration done and p has an initial value of 0.

✓ The actual output of every neuron in the hidden layer can be counted using equation 10.

$$y_j(p) = sigmoid\left[\sum_{i=1}^{n} x_i(p).w_{ij}(p) \right] \quad (10)$$

Where :
n = the number of input from neuron j in the hidder layer.
sigmoid = activation function of *sigmoid*

✓ The actual Output of every neuron in the output layer can be counted using equation 11.

$$y_k(p) = sigmoid\left[\sum_{i=1}^{m} x_{jk}(p).w_{jk}(p) \right] \quad (11)$$

Where :
m = the number of input in neuron k in the output layer.
sigmoid = activation function of *sigmoid*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3. Weight training
update of every weight in the neural network is done by applying backpropagation to the errors in the output layer.

✓ *Error* in every neuron in the output layer can be counted using equation 12.

$$\delta_k(p) = y_k(p) - y_{dk}(p) \quad (12)$$

Then the correction of the weight is counted using equation 13.

$$\Delta w_{jk}(p) = \alpha.y_j(p).\delta_k(p) - \mu.\Delta w_{jk}(p-1) \quad (13)$$

Where : α = constants which define the speed of learning from backpropagation algorithm (learning rate)
μ = constants which define the size of the weight update (momentum)

Update for every weight which is connected to the neuron in the output layer is done by using equation 14.

$$w_{jk}(p+1) = w_{jk}(p) - \Delta w_{jk}(p) \quad (14)$$

✓ *Error* in every neuron in the hidden layer can be counted using equation 15.

$$\delta_j(p) = \left[\sum_{k=1}^{m} \delta_k(p).w_{jk}(p)\right].y_j(p).\left(1 - y_j(p)\right) \quad (15)$$

Then the correction of the weight can be counted using equation 16.

$$\Delta w_{ij}(p) = \alpha.x_i(p).\delta_j(p) - \mu.\Delta w_{ij}(p-1) \quad (16)$$

Where : α = constants which define the speed of learning from backpropagation algorithm (*learning rate*)
μ = constants which define the size of weight update (momentum)

Update for every weight that is connected to the neuron in the hidden layer is done using equation 17.

$$w_{ij}(p+1) = w_{ij}(p) - \Delta w_{ij}(p) \quad (17)$$

4. Iteration

The interpolation of iteration p is as much as one and back to step 2 will be done if the error criteria is not yet as expected. The training of backpropagation algorithm is completed if the error criteria is already as expected.

As for the training parameters used for MLP in the testing with six types of hand gestures can be seen in Table 1.

Table 1: The value of MLP parameters in the traning with six types of hand gestures

Parameters	Testing number				
	1	2	3	4	5
The amount of neuron in the input layer	523	523	523	523	523
The amount of neuron in the output layer	6	6	6	6	6
The amount of hidden layer	2	2	3	3	4
The amount of neuron in the first hidden layer	250	300	200	300	300
The amount of neuron in second hidden layer	50	50	100	150	150
The amount of neuron in third hidden layer	-	-	50	75	100
The amount of neuron in fourth hidden layer	-	-	-	-	75
Learning rate	0,01	0,01	0,01	0,01	0,01
Momentum rate	0,0	0,0	0,0	0,0	0,0
Epoch	1000	1000	1000	1000	1000
Error target	0,01	0,01	0,01	0,01	0,01

Training parameter which is used for MLP in the traiing with four types of hand gestures in the research can be seen in table 2.

Table 2: The value of MLP parameters in the traning with four types of hand gestures

Parameters	Testing number				
	1	2	3	4	5
The amount of neuron in the input layer	365	365	365	365	365
The amount of neuron in the output layer	4	4	4	4	4
The amount of hidden layer	2	2	3	3	4
The amount of neuron in the first hidden layer	250	300	200	300	300
The amount of neuron in second hidden layer	50	50	100	150	150
The amount of neuron in third hidden layer	-	-	50	75	100
The amount of neuron in fourth hidden layer	-	-	-	-	75
Learning rate	0,01	0,01	0,01	0,01	0,01
Momentum rate	0,0	0,0	0,0	0,0	0,0
Epoch	100	100	100	100	100
Error target	0,01	0,01	0,01	0,01	0,01

3. RESULT AND EXPLANATION

Before we go to the result, firstly we need to explain the experimental setup which includes the hardware and software setups. For hardware specification we use Prosesor *Intel® Core™* i7-2640M *CPU @* 2.80 *GHz*, RAM Memory 4 GB DDR3, and for image or video acquisition we use builtin camera 1 MP with 1280x720 native high definition. In term of

Journal of Theoretical and Applied Information Technology
15th June 2018. Vol.96. No 11

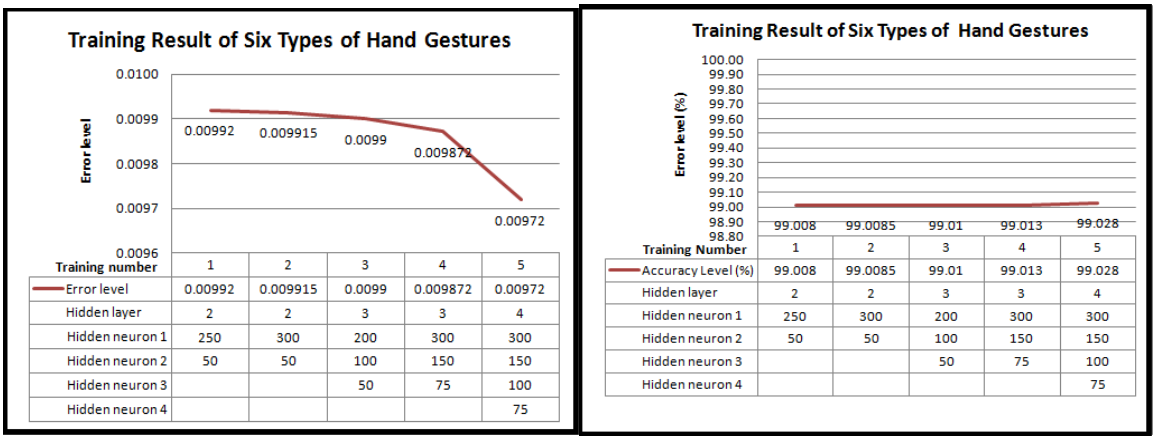
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

software we are using Eclipse IDE Mars 2. Release (4.5.2) and OpenCV 3.0.1, Encog 3.3.0, JavaCV 1.1 and OjAlgo 35.0.

3.1. Training Results

In this section the training process obtained from Multi-Layer Neural Network using training dataset is explained. There are two experiments done in the MLP training process using six types of hand gestures and two training using four stypes of hand gestures.

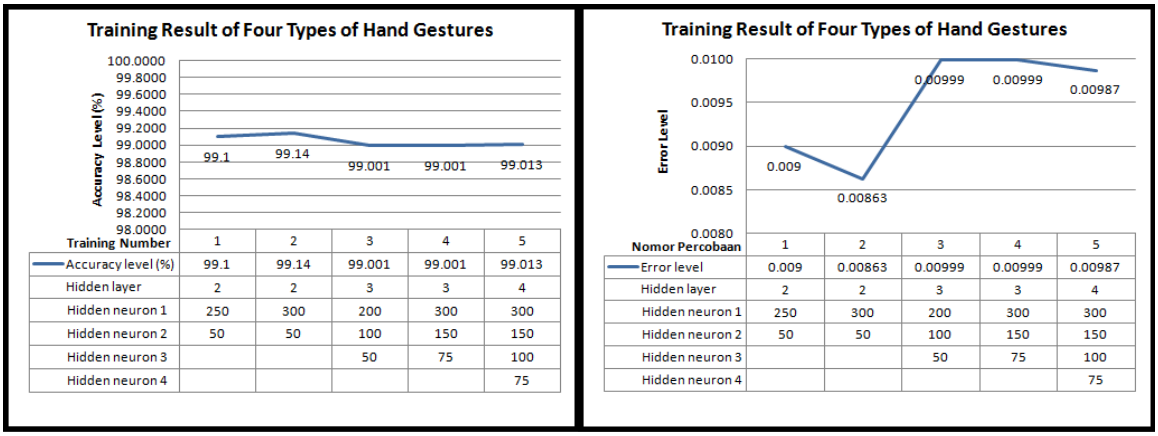
The graph representation from the training of MLP from five experiments with six types of hand gestures can be seen in Figure 2 and Figure 3. The error level of the training process in all those five experiments is lower than 0,01. From this, it can be concluded that MLP is capable of getting low error level for every amount of hidden layer and hidden neuron. All these results are obtained with the same value or learning rate and momentum rate which is 0,01 and 0,0.



Training number	1	2	3	4	5
Error level	0.00992	0.009915	0.0099	0.009872	0.00972
Hidden layer	2	2	3	3	4
Hidden neuron 1	250	300	200	300	300
Hidden neuron 2	50	50	100	150	150
Hidden neuron 3			50	75	100
Hidden neuron 4					75

Training Number	1	2	3	4	5
Accuracy Level (%)	99.008	99.0085	99.01	99.013	99.028
Hidden layer	2	2	3	3	4
Hidden neuron 1	250	300	200	300	300
Hidden neuron 2	50	50	100	150	150
Hidden neuron 3			50	75	100
Hidden neuron 4					75

Figure 2 and Figure 3. The graph resulted from the MLP training using six hand gestures



Training Number	1	2	3	4	5
Accuracy level (%)	99.1	99.14	99.001	99.001	99.013
Hidden layer	2	2	3	3	4
Hidden neuron 1	250	300	200	300	300
Hidden neuron 2	50	50	100	150	150
Hidden neuron 3			50	75	100
Hidden neuron 4					75

Nomor Percobaan	1	2	3	4	5
Error level	0.009	0.00863	0.00999	0.00999	0.00987
Hidden layer	2	2	3	3	4
Hidden neuron 1	250	300	200	300	300
Hidden neuron 2	50	50	100	150	150
Hidden neuron 3			50	75	100
Hidden neuron 4					75

Figure 4 and Figure 5. Graph from the training of MLP with four types of hand gestures

The graph representation from the training of MLP in the fifth experiment with four types of hand gestures can be seen in Figure 4 and 5. The error level of the training for every number of experiment with four types of hand gestures also showed the same result as the experiment with six types of hand gestures, which is lower than 0,01 for every amount of hidden layer and hidden neuron. The parameter value of the learning rate and the momentum rate used is the same too which is 0,01 and 0,0. The average actual epoch reached in this experiment is 32,4 as for the experiment with six types of hand gestures reached higher actual epoch which is 151.

This means that MLP can adapt faster with four types of hand gestures compared to the six types of hand gestures.

3.2. Testing Results

The testing process of MLP using testing dataset is done to know the accuracy of MLP in recognizing trained hand gestures. Testing is done on both experiments either six types or four types.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Testing is also done five times for each type for each amount of hidden layer and hidden neuron.

The result of MLP testing which have been trained for all five experiments for six types of hand gestures can be seen in the graph in figure 6 and in table 3. Overall, the most optimal result for the testing with six types of hand gestures is the second test of the experiment with two hidden layers in each and each hidden neuron is as much as 300 and 50, with an overall accuracy level of 77,02%. From the test result, it can be seen that the overmuch or lack of amount of hidden layer and hidden neuron can cause the MLP only to be able to recognize the trained dataset and not be able to recognize untrained dataset.

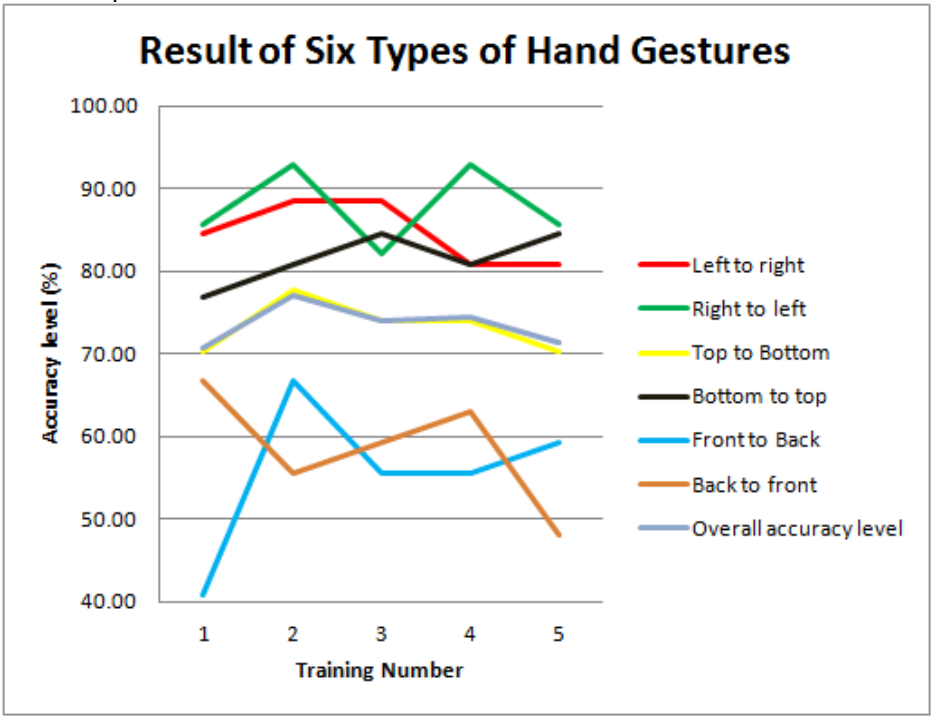


Figure 6. Graph from the testing of Six Types of Hand Gestures.

Table 3: The testing result of MLP which have been trained for six types of hand gestures

Types of hand gestures	Accuracy for number of testing				
	1	2	3	4	5
Left to right	84,62%	88,46%	88,46%	80,77%	80,77%
Right to left	85,71%	92,86%	82,14%	92,86%	85,71%
Top to bottom	70,37%	77,78%	74,07%	74,07%	70,37%
Bottom to top	76,92%	80,77%	84,62%	80,77%	84,62%
Front to back	40,74%	66,67%	55,56%	55,56%	59,26%
Back to front	66,67%	55,56%	59,26%	62,96%	48,15%
Overall accuracy level	70,81%	77,02%	73,91%	74,53%	71,43%

The testing result of trained MLP for all five experiments with four types of hand gestures can be seen in figure 7 and in table 4. Overall, the best result for the testing of trained MLP with four types of hand gestures is in experiment number one with two hidden layer and the amount of hidden neuron each is 200 and 50. The result of the testing obtained an accuracy level of 89.72% from this testing process can be concluded that the amount of second hidden layer is the optimal amount and the more amount of hidden layer could cause overfit where MLP only be able to recognize trained dataset, and not being able to recognize untrained dataset.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

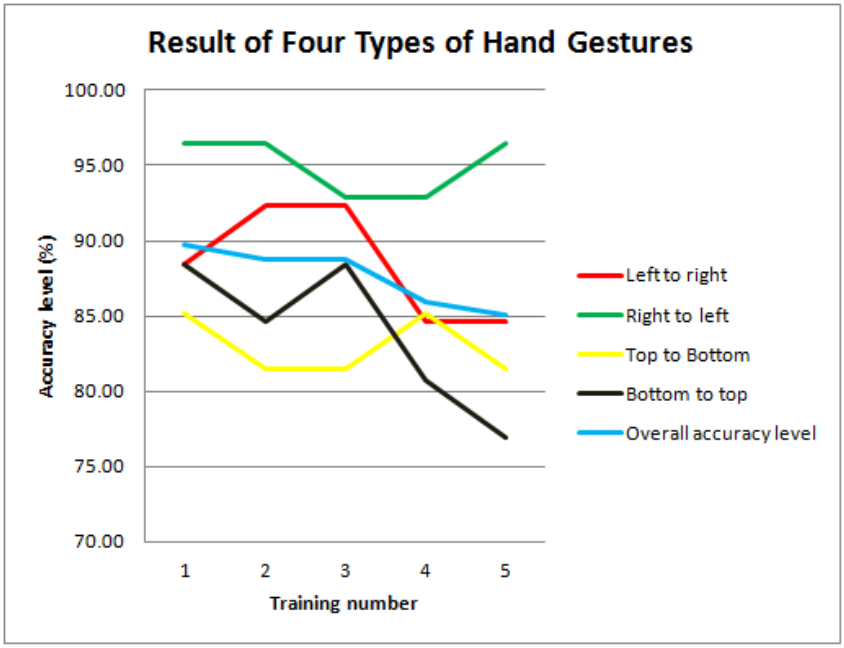


Figure 7. Graph from the testing of Four Types of Hand Gestures

Table 4: The testing result of MLP which had been trained for four types of hand gestures

Types of hand gestures	Accuracy for number of testing				
	1	2	3	4	5
Left to right	88,46%	92,31%	92,31%	84,62%	84,62%
Right to left	96,43%	96,43%	92,86%	92,86%	96,43%
Top to bottom	85,19%	81,48%	81,48%	85,19%	81,48%
Bottom to top	88,46%	84,62%	88,46%	80,77%	76,92%
Overall accuracy level	89,72%	88,79%	88,79%	85,98%	85,05%

Generally, the accuracy result of the testing with four types of hand gestures is much better than the six type's one. From both experiments, the best result obtained is from the first experiment for four types of hand gestures with two hidden layers and 200 and 50 hidden neurons with an accuracy of 89,72%. Parameters and the result of MLP obtained from this experiment then will be implemented into the front-end system to control some aplication with hand gestures.

If we compare our result with the previous work from Molchanov, we can see that the accuracy in fine tune MLP would catch CNN, however if the type of the hand gesture increases MLP can be trapped in overfitting problem. However in our case to give the movement to do activity for people with finger disability, four movement is quite enough.

4. APPLICATION

This section lists some applications which are suitable for running the hand gesture program:

4.1. File explorer

Details of procedures to control the *file explorer* using hand gesture to replace the function of a *mouse* or *keyboard* can be seen in table 5.

Table 5. Hand Gestures Function in *File Explorer*

No.	Types of hand gesture	Function
1	Right to left	To the previous folder
2	Left to right	To open a *file* or folder
3	Top to bottom	To move down
4	Bottom to top	To move up

And the UI when the file explorer is running can be seen in figure 11.

Journal of Theoretical and Applied Information Technology
15th June 2018. Vol.96. No 11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

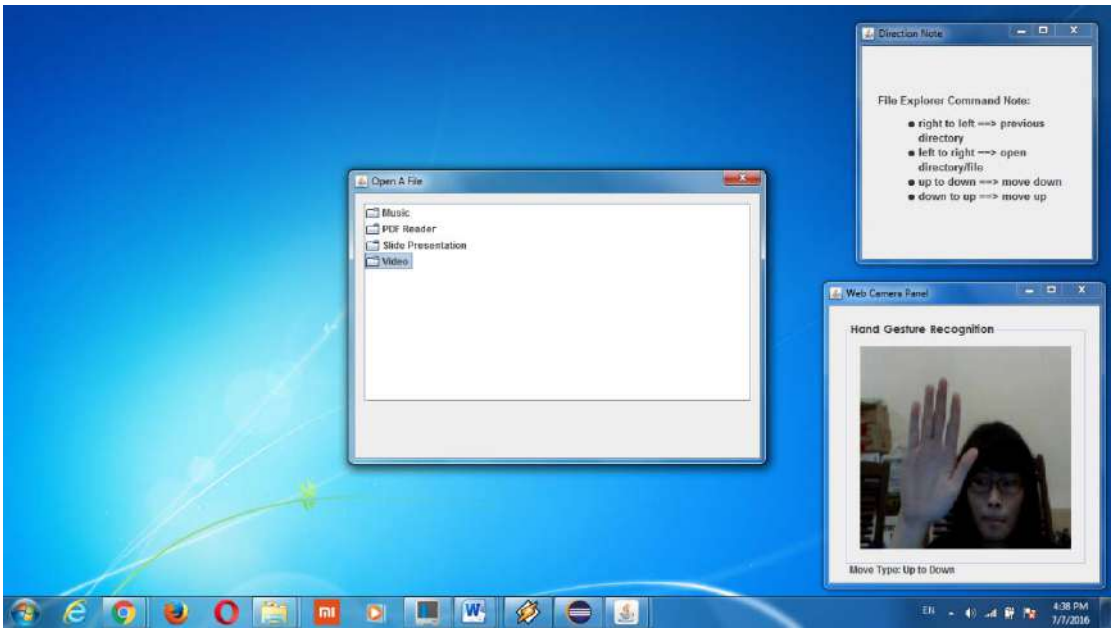


Figure 11. File Explorer UI

4.2. Windows Media Player

Music files with .mp3 extension can be opened using *Windows Media Player*. Details of procedures to control the *Windows Media Player* using hand gesture can be seen in table 6.

And the UI when *Windows Media Player* is running can be seen in figure 12.

Table 6: Hand Gestures Function in Windows Media Player

No.	Types of hand gestures	Function
1	Right to left	To lower the volume
2	Left to right	To increase the volume
3	Top to bottom	To *play* or *pause*
4	Bottom to top	- *Stop* (when the music is being played or *paused*) - To close the application (when the music is stopped/*stop*)

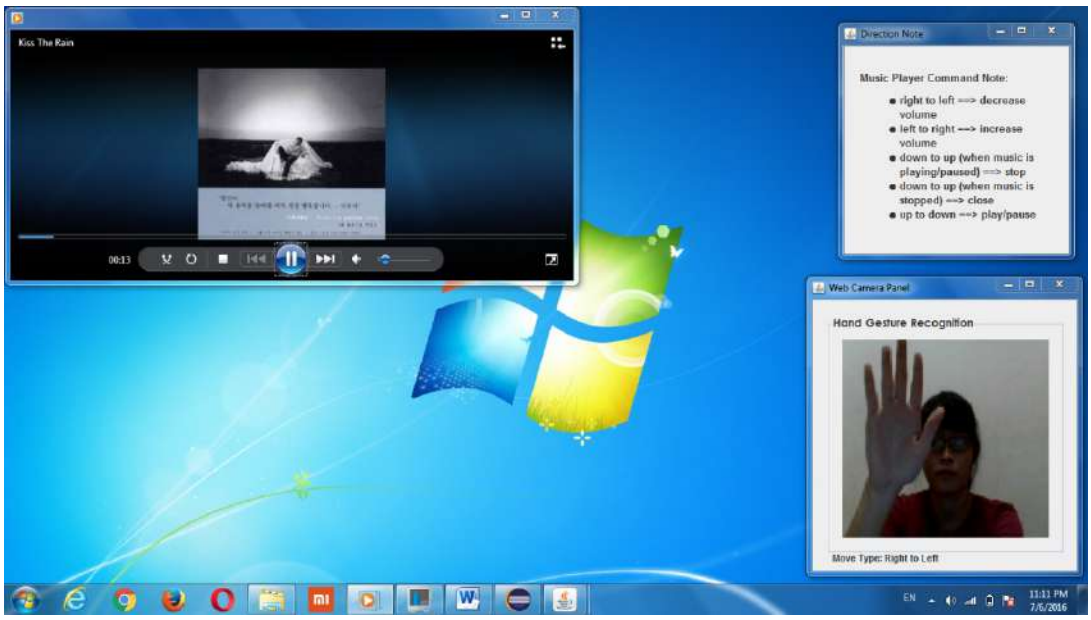


Figure 12. Windows Media Player UI

Journal of Theoretical and Applied Information Technology
15th June 2018. Vol.96. No 11

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4.3. VLC media player

Video files with .avi, .flv, .mkv, .wmv, .mp4 extension can be opened using *VLC media player*. Details of hand gestures that can be done to control the *VLC media player* can be seen in table 7.

And the UI when *VLC media player* is running can be seen in figure 13

Table 7: Hand Gestures Function in VLC Media Player

No.	Types of hand gestures	Function
1	Right to left	To lower the volume
2	Left to right	To increase the volume
3	Top to bottom	To *play* or *pause*
4	Bottom to top	- *Stop* (when the video is being played or *paused*) - To close the application (when the video is stopped/*stop*)

.

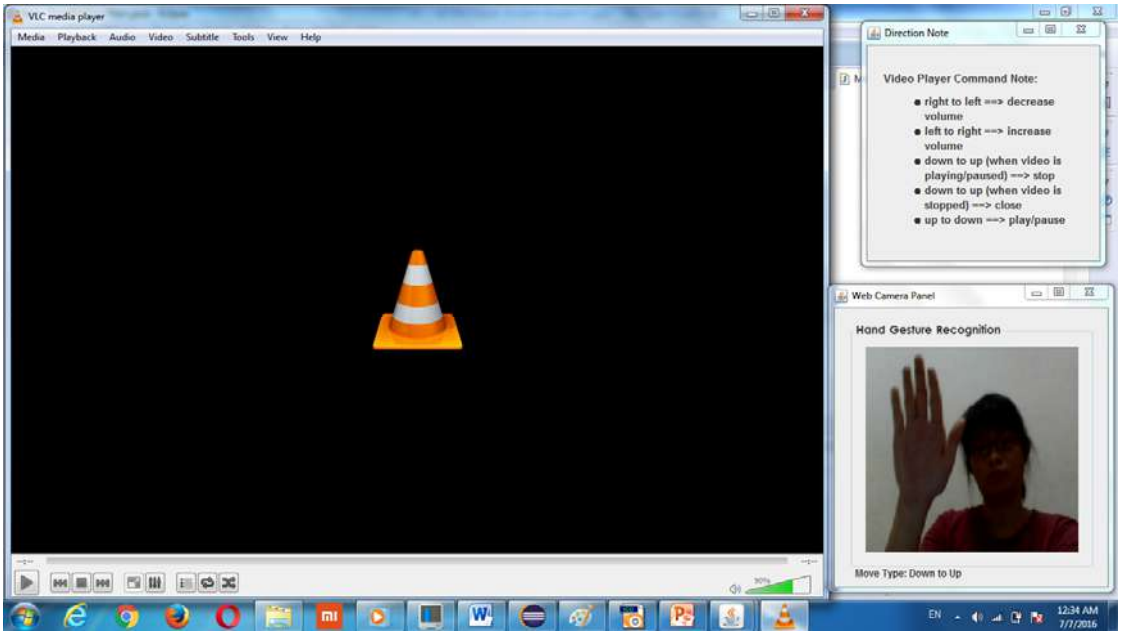


Figure 13. VLC media player UI

4.4. Microsoft Power Point

Presentation files with .ppt or .pptx extension can be opened using *Microsoft Power Point*. Details of hand gestures that can be done to control the *Microsoft Power Point* can be seen in table 8.

And the UI when *Microsoft Power Point* is running can be seen in figure 14.

Table 8: Hand Gestures Function in Microsoft Power Point

No.	Types of hand gestures	Function
1	Right to left	To the previous slide
2	Left to right	To the next slide
3	Top to bottom	*Slide show*
4	Bottom to top	- *End show* (when *slide show*) - To close the application (when *end show*)

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

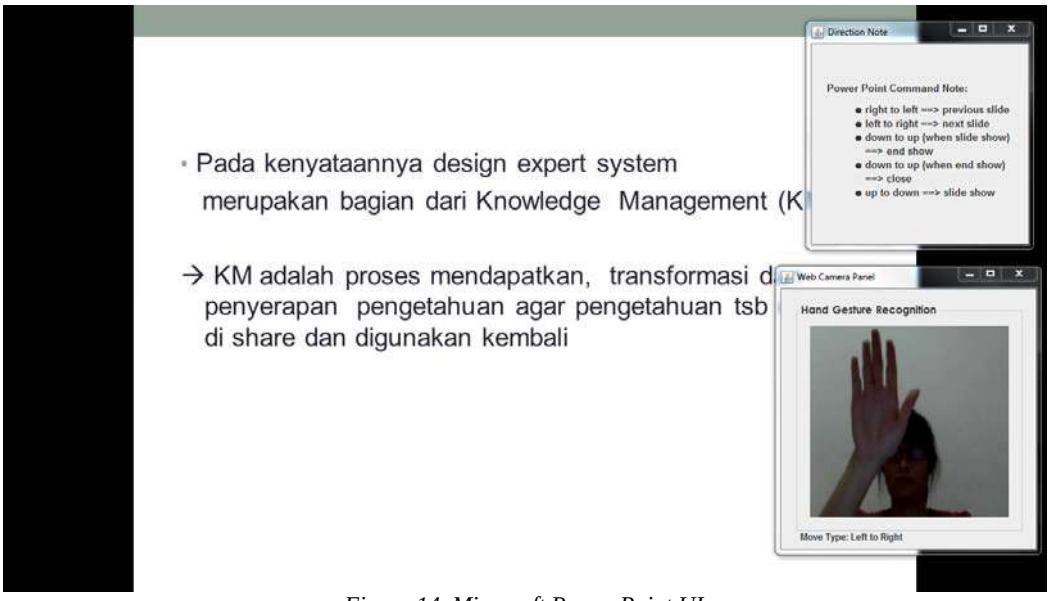


Figure 14. Microsoft Power Point UI

4.5. Acrobat Reader DC

PDF files in this reserach can be opened using *Acrobat Reader DC.* Details of hand gestures that can be done to control the *Acrobat Reader DC* using hand gestures can be seen in table 9.

And the UI when *Acrovat Reader DC* is running can be seen in figure 15.

Table 9: Hand Gesture Function in Acrobat Reader DC

No.	Types of hand gestures	Function
1	Right to left	To close the application
2	Left to right	*Zoom in* or *zoom out*
3	Top to bottom	*Scroll down*
4	Bottom to top	*Scroll up*

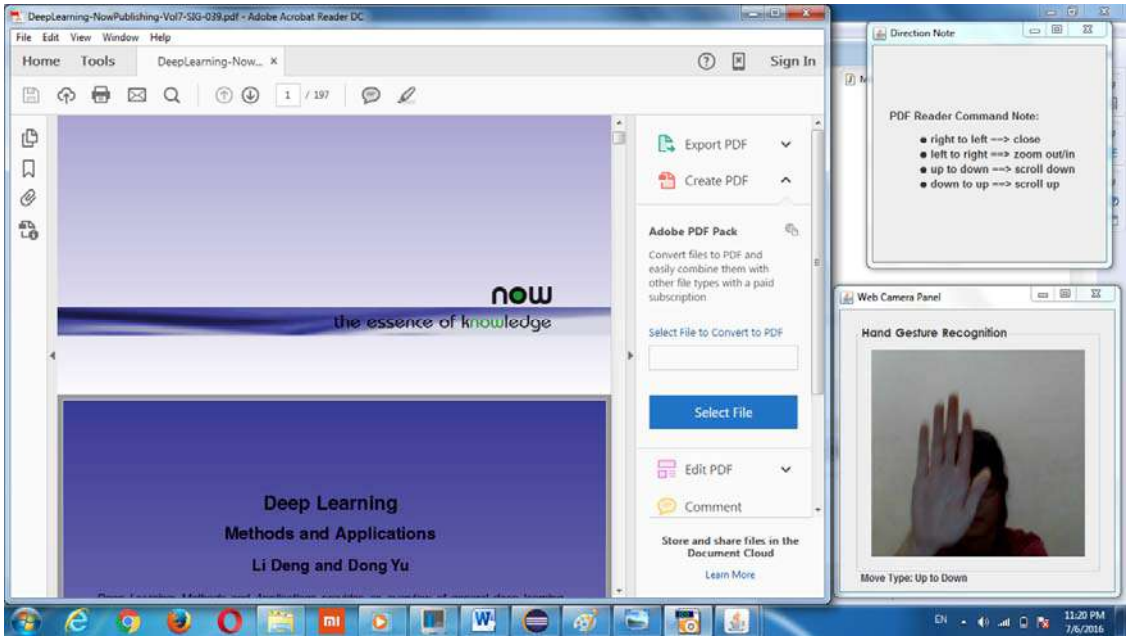


Figure 15. Acrobat Reader DC UI

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5. CONCLUSION

Based on the applied method from the proposed general architecture designed to recognize human hand gestures for Human and Computer Interaction, it can be concluded that the proposed method is able to recognize human hand gestures for Human and Computer Interaction. The highest accuracy level in the experiment with six types of hand gestures is in experiment number two with two hidden layers and the amount of each hidden neuron is 300 and 50 with an accuracy percentage of 77,02%. The accuracy level in the experiment with four types of hand gestures is in experiment number one with two hidden layers and the amount of hidden neuron is 250 and 50 with an accuracy percentage of 89,72%. The front-end system of this research implemented MLP in experiment number one with four types of hand gestures. The established front-end system can control some applications such as *file explorer* for navigating with files, *Windows Media Player* for music files, *VLC media player* for video files, *Microsoft Power Point* for presentation files and *Acrobat Reader DC* for PDF files. The limitation of this work is based on the platform that has been use, no parallel performance applied since it is using CPU resource, and the classifier needs more improvement.

For the future research it is suggested that image processing method be applied to process the video into image files which can then be used as an input parameter for the dataset to train and test the MLP so that it can further represent video data better, adding more applications that can be controlled by the front-end system with recognized hand gestures so that it will help the handicapped with more hand gestures variation, and lastly using Graphical Processing Unit (GPU) to get improvement in term of performance.

ACKNOWLEDGEMENT

This research is supported by Universitas Sumatera Utara under TALENTA 2017 Penelitian Terapan's Grant. Special thanks to Rector of Universitas Sumatera Utara, Prof. Dr. Runtung Sitepu and Head of Lembaga Penelitian, Prof. Dr. Erman Munir, for their trust and financial support.

REFERENCES

[1] Pedro Neto, Dário Pereira, J Noberto Pires, A. Paulo Moreira, "Real-time and continuous hand gesture spotting: an approach based on artificial neural networks", *IEEE International Conference on Robotic and Automation (ICRA)*, 2013, pp. 178-183.

[2] Momin Rijwan Ramjan, Rane Mithilesh Sandip,Phatangare Sushant Uttam, Wani Sumit Srimant, "Dynamic hand gesture recognition and detection for real time using human computer interaction", *International Journal of Advance Research in Computer Science and Management Studies (IJARCSMS)*, 2014, 2(3): 425-430.

[3] Romi Fadillah Rahmat, Tengku Chairunnisa, Dani Gunawan, Opim Salim Sitompul, "Skin color segmentation using multi-color space threshold ", in *2016 3rd International Conference on Computer and Information Sciences, ICCOINS 2016 - Proceedings*, 2016, pp. 391–396.

[4] Li Deng, Dong Yu, "Deep Learning Methods and Applications" 978-1-60198-814-0, Now: Netherland, 2014.

[5] Ao Tang, Ke Lu, Yufei Wang, Jie Huang, Houqiang Li, "A real-time hand posture recognition system using deep neural networks", *ACM Transactions on Intelligent Systems and Technology*, 2013, 9(4): 39:1-21.

[6] Dumitru Erhan, Christian Szegedy, Alexander Toshev, Dragomir Anguelov, "Scalable object detection using deep neural networks", *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2155-2162.

[7] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, Jan Kautz, "Hand gesture recognition with 3D convolutional neural networks", *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 1-7.

[8] Hazarathaiah Malepati, "Digital Media Processing", 978-1-85617-678-1, Elsevier: Burlington, 2010.

[9] Gary Rost Bradski, Adrian Kaehler, "Learning OpenCV", O'Relly Media, Inc: Sebastopol, 2008.

[10] D. Stalin Alex, A. Wahi, "BFSD: Background subtraction frame difference algorithm for moving object detection and extraction", *Journal of Theoretical & Applied Information Technology*, 2014, 60(3): 623-628.

[11] I.T. Jolliffe, "Principal Component Analysis", Springer: London, 2002.

[12] Michael Negnevitsky, "*Artificial Intelligence: A Guide to Intelligent Systems*", 2nd Edition. Pearson Education Limited: Upper Saddle River, 2005.